**FIG. 1**

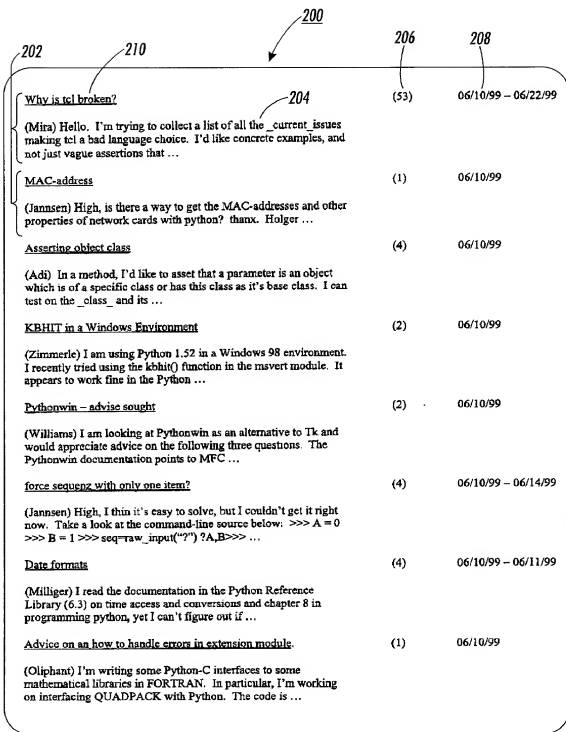


FIG. 2

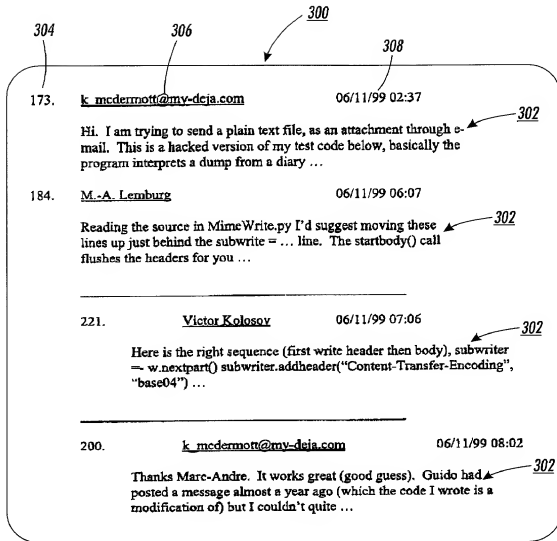


FIG. 3

723. Mark Hammond

06/18/99 17:15

[Schulenburg: I have been using a freeware product called AutoDuck to document my Python source code.]

Cool! I use AutoDuck to document all my Python extension C++ code. All the helpfiles in the Win32 extensions are generated using autodock.

Sorry in advance for the length of this, but I am in exactly the same predicament as you, and have been thinking for a number of years how to resolve this. I'm really pleased to finally find someone else with this problem, and really hope we can solve it in a reasonable way. ...

However, depending on how far you have gone with AutoDuck, pythondoc/gendoc isn't solving the exact same problem.

IMO, AutoDuck generates far superior reference documentation.

A new release of pythondoc uses XMI. I believe, so PythonDuck could be a tool that can enhance the documentation generated by pythondoc.

But in summary, Pythondoc exists and is very cool, but doesn't quite cover what AutoDuck can do (and vice-versa, but that isn't the point now). I believe a real need exists for the functionality AutoDuck provides, and further I need it personally!

**FIG. 4**

723. Mark Hammond

06/18/99 17:15

[Schulenburg: I have been using a freeware product called AutoDuck to document my Python source code.]

Cool! I use AutoDuck to document all my Python extension C++ code. All the helpfiles in the Win32 extensions are generated using autoduck.

As another poster pointed out, you should look at pythondoc/gendoc. Sorry in advance for the length of this, but I am in exactly the same predicament as you, and have been thinking for a number of years how to resolve this. I'm really pleased to finally find someone else with this problem, and really hope we can solve it in a reasonable way.

However, depending on how far you have gone with AutoDuck, pythondoc/gendoc isn't solving the exact same problem. Well, they all attempt to solve the same problem, but have different strengths.

IMO, AutoDuck generates far superior reference documentation. Pythondoc's big strength is the integration with Python.

AutoDuck as 3 strengths over pythondoc that I am having trouble reconciling: pythondoc really doesn't support natural links to objects. Autoduck can say "see method" or "@xref objectmethod" to generate inline links or a cross-references section. Has been some discussion how to expand Pythondoc to support this, but a) it hasn't been done and b) it still is not as natural as autoduck.

I have come to the conclusion that we need a new tool - PythonDuck. It should work with Pythondoc, and be capable of enhancing its output. A new release of pythondoc uses XML. I believe, so PythonDuck could be a tool that can enhance the documentation generated by pythondoc. Alternatively we just graft it in, and beat up Daniel until he agrees :-)

This tool could possibly still work from comments, or maybe from the tad-end of the docstrings (thereby stripped from the documentation, and not messing the raw docstring too much). #@ comments could be considered "directives" for the generated output. Eg. "#@ xref SomeOtherFunction" will create a cross-references section. A comment such as "#@ paramtype param\_name" would direct that the param called "param\_name" should be documented as being of type "SomeType" and a cross-reference to the SomeType definition be added. The tool should resolve all links at the end, and report bad links.

I know I am not making my point that well, and unfortunately don't have time to fix this. But in summary, Pythondoc exists and is very cool, but doesn't quite cover what AutoDuck can do (and vice-versa, but that isn't the point now). I believe a real need exists for the functionality AutoDuck provides, and further I need it personally! So we should try and make this happen.

Mark.

FIG. 5



606

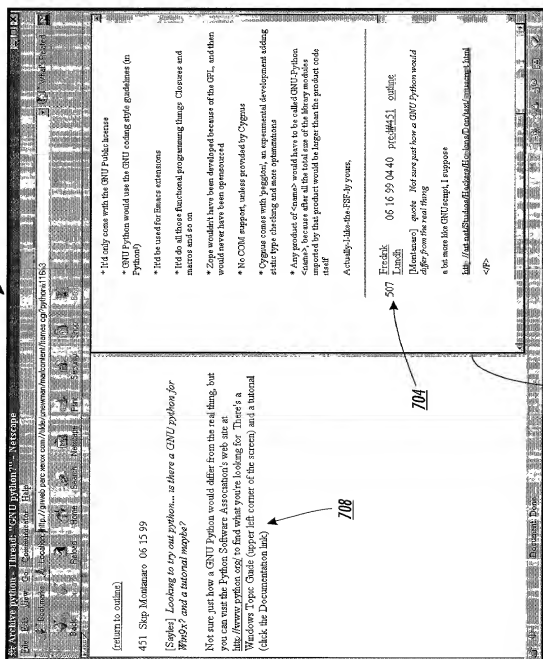


FIG. 7

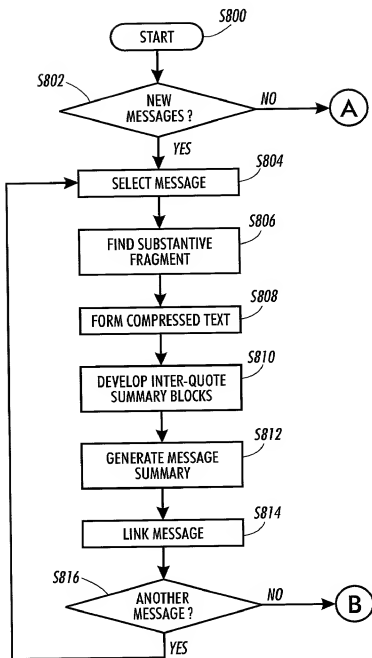


FIG. 8



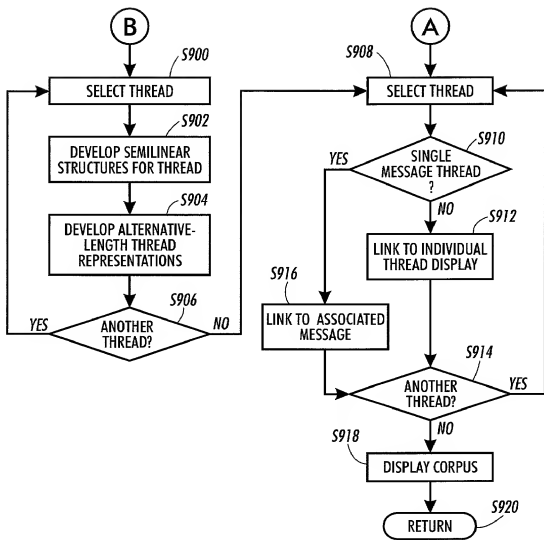


FIG. 9